TWR-MCF5225X

TOWER SYSTEM



Connectivity Labs 5 and 6

Functional Elevator

streescale

semiconducto

About the Tower System

Lab Tutorials for TWR-MCF5225X



Get to know the TWR-MCF5225X





TWR-MCF5225X-KIT **Freescale Tower System**

The TWR-MCF5225X module is part of the Freescale Tower System, a modular development platform that enables rapid prototyping and tool re-use through reconfigurable hardware. Take your design to the next level and begin constructing your Tower System today.

TOWER SYSTEM

MCF5225X—Lab Tutorials 5 and 6 (sheet 3 of 3)



Finding an Error Using Task Aware Debugging (TAD) in CodeWarrior™

This lab will show you the power of using task-aware debugging to troubleshoot your application. An error has purposely been introduced into this project, and this lab will show you how to find and solve that error.

Demonstrates

- Sending messages between tasks
 (logging task)
- CodeWarrior task-aware debugging windows

Step by Step Instructions

- 1. Stop the application if it is currently running (**Debug>Kill**)
- Open the Lab Project by selecting the File > Open menu item: C:\Program Files\FreescaleMQX3.4\ demo\hvac_error\codewarrior\ hvac_error_twrmcf52259.mcp
- 3. Enable the auto logging feature. This is done by opening the **hvac.h** file
- 4. Then change the auto logging define:
 #define DEMOCFG_ENABLE_AUTO_LOGGING 0
 to this:
- #define DEMOCFG ENABLE AUTO LOGGING 1

hvac	_m52259evb.mcp					
Files	HVAC - Int. Flash Release 💌	10 🖋	Ø \$	Ø 🖡 🕨		
*	File	Code	Data	0	*	
	🗋 Linker Files	0	0			-
	🗎 Runtime Libraries	234K	12129	•		-
	MQX Libraries	113K	4391			-
	RTCS Libraries	249K	23123	٠		-
E-C	MFS Libraries	24322	1045			-
+	USB Host Libraries	27060	676	•		-
+-(Shell Libraries	27684	28403			-
Be	Source	5618	3030			-
	+ 🗀 Shell	1508	1438			-
1	🗄 🦳 HVAC	2154	581			-
	- hvac.h	0	0			-
	HVAC IO.c	616	145			-
	hvac private.h	0	0			-
	- hvac public.h	0	0			-
	HVAC Task.c	254	26			-
	HVAC UtiLc	594	137			
	Logging Task.c	394	50	٠		-
	logging private h	0	0			-

- Notice that the impacted files have a red check mark beside them to indicate that they need to be re-compiled, or are "touched." Compile, download, and run the application as was done in steps 8 to 15 of Lab 1.
- Go to the shell console in hyperterminal. Note that a string of logging information will be printed out every 15 seconds. It will also be printed out when there is an update to any of the parameters such as the desired temperature. You can test this by pressing SW1.
- Press SW1 until the desired temperature gets to 24°C and then use SW3 to bring it back down to 20°C. You should notice that the logging will eventually stop and that no more updates are printed.

Dell				_				_			
inell> inell> mode: 0 mode: 0	node ####################################	t: Off, set:69. set:70. set:70. set:71. set:71. set:72. set:73. set:73. set:73. set:73. set:73. set:73. set:73. set:74. set:72. set:72. set:72. set:72. set:72. set:72. set:72. set:72. set:72. set:72. set:72. set:72. set:72. set:72. set:73. set:74	set:68.5 tr 0 temp:68.4 0 temp:68.4 m displayed of temp:68.4	tep:66 9 Fan: 9 Fan:	O Fan Auto Auto Auto Auto Auto Auto Auto Auto	: Auto F Fan Off, Fan Off,	an OFF, Furnace Furnace Furnace Furnace Furnace Furnace Furnace Furnace Furnace Furnace	urna Off, Off, Off, Off, Off, Off, Off, Off	te DFF, A A/C DFF, A/C OFF, A/C OFF,	U/C OFF,	

Figure 2: Up to 24, on the way down it stops logging

 Your job now is to use the task aware debugging (TAD) feature to check for errors to determine why this error is happening. To see the TAD data, pause the application by clicking on the Break icon (which is a red square) or by selecting Break in the Debug menu.



Figure 3: Break application execution

Kernel Data	
Check for Errors	8
CHECK IOI LITOIS	-11
Task Summary	- 84
Ready Queues	- 84
Stack Usage	
Memory Pools	
Memory Blocks	- 84
Memory Extension Blocks	- 84
Lightweight MemoryPool	- 84
Partition Summary	
Message Queues	
Message Pools	- 84
Lightweight Message Queues	
Lightweight Events	-
Lightweight Semaphores	- 84
Events	- 84
Mutexes	- 84
Semaphores	- 84
Task Queues	
Initialization	
Interrupt Summary	- 84
IO Devices	- 84
Lightweight Timer Queues	- 84
Kernel Log	- 84
Log Summary	
Names	
MQX Data	-
Performance Data	- 88
Close All	
Help	
About	- 88

MOX RTCS Tools Processor Expert

Figure 4: MQX Task-Aware Debugging menu

- Then click on the MQX pull down menu. Read down the list of available information windows and select the one(s) that you might think would be a good indication as to what happened.
- 10. One of the TAD windows which will quickly help you to get an application status overview is Task Summary or Check for Errors.

🖬 Task Summary					
Task Name	Task ID	TD	Priority	State	Task Error Code
mgx idle task	0×10001	0x20001ec0	13	Ready	OK
HVAC	0×10002	0x20002100	9	LW Event Blocked, timeout	MSGP: Out of Messages
Switch	0x10003	0x200027f0	10	Time delay blocked	OK
Shell	0×10004	0x20002cf0	12	Active	OK
Logging	0×10005	0x20003da0	11	Rx Msg Blocked	OK

- [D] ×

Figure 5: Task Summary TAD window

Solution

The HVAC task is using MQX messages to send data to the Log task (see HVAC_ LogCurrentState() and the Log() functions) in **HVAC_Util.c** and **Logging_Task.c**. The HVAC task, as the message sender, assumes each message is "consumed" by the Log task and removed from the message pool after the text is printed to the console.

From the HVAC task error code found in the Task Summary TAD window seen in Figure 5, it is apparent that a message could not be sent because the message pool is full. You can verify this assumption by showing the Message Pools TAD window (Figure 6) and double-clicking on the one and only message pool entry to bring up the window shown in Figure 7.

rssage Pool Summary /ID Size Total# Free Limit

Figure 6: Message Pool Summary TAD window

Examine Message	Pool		_ ×		
Pool ID:	0×20004bb0	Msgs in Pool:	16		
Pool Type:	Private Pool	Free Msgs:	0		
Message Size:	0×88	Grow Number:	0		
Header:	Valid	Grow Limit:	16		
Msg Address	State	Details			
0x20004d74	Owned	Task 0x10005			
0x20004e14	Owned	Task 0×10005			
0×20004cb4	Owned	Task 0×10005			
0x20004f54	Owned	Task 0×10005			
0×20004ff4	Owned	Task 0x10005			
0×20005094	Owned	Task 0x10005			
0×20005134	Owned	Task 0×10005			
0x200051d4	Owned	Task 0x10005			
0×20005274	Owned	Task 0×10005			
0×20005314	Owned	Task 0×10005			
0×200053b4	Owned	Task 0×10005			
0×20005454	Owned	Task 0×10005			
0x200054f4	Owned	Task 0x10005			
0×20005594	Owned	Task 0×10005			
0×20005634	Owned	Task 0×10005			
0×200056d4	Owned	Task 0×10005			
4			1000		

Figure 7: Message pool examined (after double-clicking the pool entry) The message pool is exhausted. The problem is on the receiving side, as it is always the message receiver's responsibility to reuse the message object or free it when no longer needed.

Looking at the Logging_task() function, located in **Logging_Task.c**, you can see the message is received by

msg_ptr = _msgq_receive(log_qid, 0);
and after that the data of the message
(the log text) is printed

printf(msg_ptr->MESSAGE);

What is missing is deletion of the message after the log text is printed:

_msg_free(msg_ptr);

Add this line: recompile and run the application. The message memory will now be released after the message is printed out.



Figure 8: Correct code



Ethernet to Serial Bridge, Freescale MQX RTCS

This lab demonstrates how to create a bridge between a TCP/IP (telnet) connection and a serial line.

Demonstrates

- MQX RTCS TCP/IP network stack
- Custom telnet server implementation
- Re-directing STDIN and STDOUT output within an MQX task

Step by Step Instructions

- 1. Make the following connections from the Tower System to the computer.
 - a. USB debugger connection (J17 on the TWR-MCF5225X module) to a USB port on PC
 - **b.** Serial port on the **TWR-SER** module to a serial port on PC (serial cable not included)
 - c. An Ethernet cable between the TWR-SER module and an Ethernet port on your computer
- The first time you connect the USB debugger cable to your PC, Windows will install a driver for the debugger. Follow the prompts to automatically detect and install the driver.
- 3. Open the lab project by selecting the File > Open menu item: C:\Program Files\FreescaleMQX3.4\ demo\telnet_to_serial\codewarrior\ telnet2ser_twrmcf52259.mcp
- 4. The default IP address of the board is 169.254.3.3. Typically, when you connect your computer directly to the board, the computer will default to an auto IP address on the same subnet as the board (169.254.x.x), therefore requiring no setup. Note: The PC may take a few minutes to default to the auto IP address and make the connection.

continued on reverse side...





commune nom roverse side...

However, if you have trouble connecting, you may configure the IP address of the computer manually. Select Start > Settings > Network Connections > Local Area Connection. Note your original TCP/IP settings, and then set your IP address to 169.254.3.4 and your subnet mask to 255 255 0.0

۲	Telnet2Ser • Int. Flash D 💌 🚺	a 😽 🤇	\$ 💺	▶ [1		
Files	S Link Order Targets						
*	File		Code	Data	10	*	
¥ +	🗀 Linker Files		0	0	٠		-
# +	Runtime Libraries		0	0			1
* +	MQX Libraries		0	0			-
* +	ATCS Libraries		0	0			-
¥ =	C Source		0	0	•		1
	config.h		0	0			-
*	demo.c		0	0			1
*	📓 lw_telnet.c		0	0	•	•	-
	I lu telpet b		0	0			-

Figure 1: MQX source tree

 Open the config.h file in the CodeWarrior window as shown in Figure 1. Double click the file item located in the "Source" group in the CodeWarrior project tree. Locate the line of code starting with #define ENET_IPADDR and specify your target IP address by using the IPADDR macro. Set the target address to 169.254.3.3, and the line will be:

#define ENET_IPADDR IPADDR(169,254,3,3)7. Do the same with the IP address mask value ENET_IPMASK:

#define ENET_IPMASK IPADDR(255,255,0,0)

- Compile, download, and run the application and open a hyperterminal window as was done in steps 8 to 15 of Lab 1.
- Open a command prompt on the PC (Start > All Programs > Accessories > Command Prompt). At the prompt invoke a telnet session to the board by typing telnet 169.254.3.3 You will be connected to the MQX shell via telnet.
- 10. Now the serial console and the telnet sessions should be "bridged." Type some characters into the telnet session, and you should see the characters appearing on the console terminal window. See Figure 2 for how it will appear.

from hyper	terminal window
	te Edit View Call Transfer Help
) 🖨 🖉 🕲 🗳 🗳
	Turned from toland minder
	Typed from ternet window

Figure 2: Ethernet to Serial Bridge

 Then try typing into the console terminal window, and you should see the characters appearing in the telnet session.

TOWER SYSTEM

TOWER SYSTEM

Learn More: For more information about MQX and Freescale solutions, please visit www.freescale.com/mqx and www.freescale.com/tower.

Freescale and the Freescale logo are trademarks or registered trademarks of Freescale Semiconductor, Inc. in the U.S. and other countries. All other product or service names are the property of their respective owners. © Freescale Semiconductor, Inc. 2009.



Doc Number: TWRMCF52259LAB3 / REV 0 Agile Number: 926-78390 / REV A